IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION

OF

LAURENS LEURS

AND

KOEN VAN DE POEL

FOR

METHOD FOR OUTPUTTING A DIGITAL PRODUCT DEFINITION OF A

PRINTED PRODUCT TO A LAYOUT APPLICATION

METHOD FOR OUTPUTTING A DIGITAL PRODUCT DEFINITION OF A PRINTED PRODUCT TO A LAYOUT APPLICATION

The application claims the benefit of US Provisional
5   Application No. 60/428,837 filed on November 25, 2002 and of US
Provisional Application No. 60/451,405 filed on March 4, 2003.

FIELD OF THE INVENTION

10   The present invention relates to the generation of a digital
representation of a product that is to be printed.

BACKGROUND OF THE INVENTION

15

In the printing and publishing environment, different players
interact in order to obtain print and publishing products, such as
magazines, catalogues, promotional, corporate, book or specialty
products in offset, flexo, screen, digital, sheet- or web-fed
20   printing. Such products are called "printed products" in this
document.
First, some terms are defined that are useful in the present
patent application.
"Digital Representation of a Printed Product": the product that
25   is to be printed, represented in digital form.
"Artwork": the electronic document that is generated by the
"designer", created while taking into account the "product
definition". The artwork is also called design, layout or
publication. The artwork is thus a particular digital
30   representation of the printed product.
"Designer": the person in the creative department who will
create an artwork in electronic format based on specific
instructions and content.
"Product Definition": the definition for the final printed
35   product, holding manufacturing specifications. Some specifications

that are important for a designer are: page size, bleed size, number
of pages, set of colors used. A product definition often starts
from a product intent.

"Layout application": a software application that is used by
5   the designer to generate the artwork. Common layout applications
are InDesign^{TM} from Adobe, Xpress^{TM} from Quark, Illustrator^{TM} from
Adobe, FreeHand^{TM} from Macromedia.

"Pre-press workflow application": the pre-press department sets
up a pre-press workflow for the printed product; the pre-press
10 workflow application is the software application that is used to
generate the pre-press product. Agfa Apogee^{TM} is an example of a
pre-press workflow application.

The designer needs to create the artwork according to the
product definition. Later, the pre-press department will set up a
15 workflow system, also according to the product definition. A
problem is that, after the pre-press department receives the artwork
from the creative department, and proceeds with it (e.g. Agfa
Apogee^{TM} may receive a PDF-file from Adobe InDesign^{TM} - wherein PDF
stands for Portable Document Format), errors may be found. Such an
20 error may be due to a mistake of the designer, who applied the
instructions he received to create the artwork in a wrong way.

This problem is especially important since the creative
department and the pre-press department are really "two different
worlds" ("creative" people versus "production" people). Often, the
25 creative and the pre-press departments belong to different
companies.

There is still a need for an improved method for making a
digital representation of a printed product.

30

SUMMARY OF THE INVENTION

The present invention is a method for making a digital
representation of a printed product, as claimed in independent
35 claims 1 and 15. Preferred embodiments of the invention are set out

in the dependent claims. Preferably, a method in accordance with
the invention is implemented by a computer program product as
claimed in independent claims 10 and 22. The invention also
includes a data processing system that is suitable for carrying out
a method in accordance with the invention, and a computer readable
medium comprising program code adapted to carry out a method
according to the invention when the program code is run on a
computer.

Now, some additional terms are defined that are useful in the
present patent application.

"Planning Application", also called "Planning system", "Project
Management System" or "MIS" (acronym for Management Information
System): a system that creates the product definition. In one
embodiment of the invention, this product definition is used both by
the creative department, that creates the artwork, and by the pre-
press department, that sets up the pre-press workflow for the
printed product. Generally, the product definition is also used by
the finishing and the printing departments. An example of a
planning application is the Agfa Delano^{TM} software.

"Scripting": a way to automate operations in an application.
Scripting may be used to preset and/or lock some attributes in a
layout application.

"Template": a good example is a Word template. It is a special
type of document that is set up with the proper information. When
the designer starts to create the artwork, he may ask the layout
application to create a new document based on a specific template.

For some applications, the template is limited to one (or two
facing) page(s). In this case, the exact number of pages cannot be
controlled.

A template, as defined in the present document, is not to be
confused with an Imposition template.

"Starting document": similar to a template but it is now
already a 'real' document. The application can 'Open' it directly
and start working into it.

"Template Generator": a software application that creates a template to be used by a layout application, based on certain parameters (of the product definition).

"Locked" (template or artwork): one or more crucial attributes
5 of a template or document may be locked so that the designer cannot modify them. Examples of such crucial attributes are page size, bleed size, number of pages, color model, spot colors. Remark: the terms "attributes" and "parameters", e.g. of the product definition, are used as synonyms in the present document.

10 In a preferred embodiment of the invention, a planning application is used to create the product definition, and this product definition is used by the creative department, that creates the artwork. Preferably, this same product definition is also used by the pre-press department, that sets up the pre-press workflow for
15 the printed product. This is advantageous since errors are avoided that may result from the transfer of the instructions from the planning system to the designer. Another advantage is that the pre-press department gets the instructions in the same way as the designer, which also avoids possible errors. It is preferred that a
20 central planning system, that covers both the creative department and the pre-press department, is used to define the product before the creative work is started. In this way, no errors are introduced when transferring the product definition to another department.

In a particular embodiment of the invention, the basic document
25 is then created automatically, based on this product definition. Further, it is preferred that at least one, and that more preferably all crucial attributes of the product definition are locked, so that they are protected from being modified by the layout application. Since the designer can thus no longer modify these attributes,
30 possible errors by the designer are avoided.

Further advantages and embodiments of the present invention will become apparent from the following description and drawings.

35

BRIEF DESCRIPTION OF THE DRAWINGS

The invention is described with reference to the following drawings without the intention to limit the invention thereto, and in which:

Fig. 1 illustrates an embodiment in accordance with the invention;

Fig. 2 illustrates another embodiment in accordance with the invention; and

Fig. 3 illustrates yet another embodiment in accordance with the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

As discussed already above, in a preferred embodiment of the invention a planning application is used to create a product definition, and this product definition is used by the creative department. Fig. 1 shows such an embodiment. In Fig. 1, interpreted from the point of view of a planning department, the product definition 12 is output by a planning application 10 to a layout application 20; the layout application 20 uses the product definition 12 to create an artwork 22. From the point of view of the creative department, the product definition 12 is input by the layout application 20 from the planning application 10, and the layout application 20 uses the product definition 12 to create an artwork 22.

This same product definition 12 may be output by the planning application 10 to a pre-press workflow application (not shown in the drawings) and the pre-press workflow application may then use the product definition 12 to create a pre-press product. Two practical examples are as follows:

Example 1: Agfa Delano^{TM} (as the planning application):

- creates the product definition in JDF format (JDF stands for Job Definition Format);

- sends the product definition to Adobe InDesign^{TM} (as the layout
  application);
- sends the product definition to Agfa Apogee^{TM} (as the pre-press
  workflow application).

5   Example 2: Adobe InDesign^{TM} (as the layout application):

- creates a new document according to the product definition;
- locks the attributes to avoid errors.

Remark: in example 1, "sending" the product definition from Agfa
Delano^{TM} to Adobe InDesign^{TM} means that Agfa Delano^{TM}, the planning
10 application, outputs the product definition and that Adobe
InDesign^{TM}, the layout application, inputs it. In this document,
"inputting" and "outputting" may be either directly (from the first
application to the second one) or indirectly (from the first to the
second application via an intermediate application - this is
15 illustrated e.g. by the embodiment illustrated by Fig. 3 and
discussed below.

    A method according to the invention may be implemented in many
different ways, e.g. depending on what the layout application
supports. As mentioned above, preferably a central planning system
20 is used, that covers both the creative department and the pre-press
department; such a planning system, or planning application 10, is
indicated in Figs. 1 - 3. However, a planning application 10 that
creates the product definition 12 and that only sends it to the
layout application 20, not to the pre-press workflow application,
25 may also be used in the embodiments illustrated by Figs. 1 - 3.

    Of the embodiments illustrated by Fig. 1 - 3, Fig. 1 shows the
most integrated one, followed by Fig. 2, and Fig. 3 finally shows
the least integrated embodiment.

    In Fig. 1, the planning application 10 sends the product
30 definition 12 (the product definition parameters) in a format that
the layout application 20 understands.

    Fig. 2 shows an embodiment wherein the planning application 10
sends the product definition 12 (the product definition parameters)
to a script 14, also called intermediate script 14 in the present
35 document; script 14 drives the layout application 20. The layout
application 20 is driven by script 14 to produce the starting

document (or template). Further below, an example of such a script is given for InDesign^{TM}, written in Visual Basic.

In the embodiment of Fig. 3, the product definition 12 (the product definition parameters) is sent by the planning application
5   10 to an intermediate application, a so-called template generator 15, that generates the document template 17 or starting document for the layout application 20. The intermediate application 15 translates the product definition 12 to a particular format, i.e. template 17, that the layout application understands. One or more
10 attributes of the template 17 may be locked so that the designer cannot modify them.

The following example is a script, written in Visual Basic, for InDesign^{TM}; it illustrates the embodiment of Fig. 2.
15

EXAMPLE


```
' version 0.99
' by Koen Van de Poel
' Copyright Agfa-Gevaert N.V.
' 2002-12-07
'
' Sample script for Indesign that allows to Import a JDF with
Product Intents
' set-up a UI with it and then create an Indesign or Illustrator
Document with the required settings
'
'Option Explicit
Dim xmldoc As DOMDocument


' Creates the Indesign document
Private Sub Indesign_Click()
    Dim myInDesign As Indesign.Application
    Dim myDocument As Indesign.Document
    Dim myTextFrame As Indesign.TextFrame

    Set myInDesign = CreateObject("Indesign.Application.2.0")

    Rem maak nieuwe pub
    Set myDocument = myInDesign.Documents.Add
    Rem copy from interface
    myDocument.DocumentPreferences.NumberOfPages =
CInt(NrPages.Text)
```

```
      myDocument.DocumentPreferences.PageWidth = PageWidth.Text & "mm"
      myDocument.DocumentPreferences.PageHeight = PageHeight.Text &
"mm"
      ' myDocument.Name = Title.Text
      Set myMetaDataPreferences = myDocument.MetaDataPreferences
      myMetaDataPreferences.Author = "KVDP Indesign JDF Script"
      myMetaDataPreferences.JobName = JobName.Text
      myMetaDataPreferences.Description = Description.Text
      myMetaDataPreferences.Title = Title.Text

Rem just to show something
      Set myTextFrame = myDocument.Spreads.Item(1).TextFrames.Add
      myTextFrame.GeometricBounds = Array("10mm", "10mm", "30mm",
"30mm")
      myTextFrame.TextContents = "Hello World"
End Sub
' Similar stuff for Illustrator automation
Private Sub IllustratorButton_Click()
      Dim appRef As New Illustrator.Application
      Dim newDoc As Illustrator.Document
      Dim newCMYKColor As New Illustrator.CMYKColor
      Dim newColor As New Illustrator.Color
      Dim newSpot As Illustrator.Spot
      Dim frontPath As Illustrator.PathItem

       Rem copy from interface
      'myDocument.DocumentPreferences.NumberOfPages =
CInt(NrPages.Text)
      'myDocument.DocumentPreferences.PageWidth = PageWidth.Text &
"mm"
      'myDocument.DocumentPreferences.PageHeight = PageHeight.Text &
"mm"
      ' myDocument.Name = Title.Text
      'Set myMetaDataPreferences = myDocument.MetaDataPreferences
      'myMetaDataPreferences.Author = "KVDP Indesign JDF Script"
      'myMetaDataPreferences.JobName = JobName.Text
      'myMetaDataPreferences.Description = Description.Text
      'myMetaDataPreferences.Title = Title.Text
      'AppActivate "Adobe Illustrator"
      'Set appRef = CreateObject("Illustrator.Application.10.0")

       ' Create the document
      Set newDoc = appRef.Documents.Add(aiDocumentCMYKColor,
(CInt(PageWidth.Text) * 72) / 25.4, (CInt(PageHeight.Text) * 72) /
25.4)

       ' Create a spot color (as test)
      newCMYKColor.cyan = 22
      newCMYKColor.yellow = 100
      newCMYKColor.black = 33
      newCMYKColor.magenta = 44
```

```
         newColor.CMYK = newCMYKColor

         Set frontDocument = appRef.ActiveDocument
         Set newSpot = frontDocument.Spots.Add
 5       newSpot.Color = newColor
         newSpot.ColorType = aiSpot
         newSpot.name = "Koens Spot"


     End Sub
10   Rem Import JDF
     Private Sub initxmldoc()
         Set xmldoc = Nothing
         Set xmldoc = CreateObject("Msxml.DomDocument")
         xmldoc.async = False
15   End Sub


     ' Load the JDF document into internal structure
     Private Function LoadJDF(xmldoc As DOMDocument, filepath As String)
     As Boolean
20
         Dim runlistnode As IXMLDOMNode
         Dim bLoadOK As Boolean
         Dim Title As String
         bLoadOK = False
25
         If Not (IsNull(xmldoc)) Then
             xmldoc.Load filepath ' load XML file as document
             ' quick validity check by looking for Runlist

30           If IsValidNode(xmldoc.documentElement) Then
                 Set runlistnode =
     xmldoc.documentElement.selectSingleNode(kSizeIntentDimensions)
                 If Not IsValidNode(runlistnode) Then
                     Dim mResult As VbMsgBoxResult
35                   ' allow to proceed, should be better with Proceed
     stop buttons !!
                     mResult = MsgBox("This JDF file is not supported (no
     SizeIntent found)." & Chr(10) & "Use supplied templates." & Chr(10)
     & filepath & Chr(10) & _
40                   "Click 'Yes' to Proceed anyway and 'No' to stop.", _
                     vbYesNo + vbDefaultButton2 + vbCritical)
                     If (mResult = vbNo) Then
                         bLoadDefault = False
                     End If
45               Else: bLoadOK = True
                 End If
                 If bLoadOK Then
                     ' check if file has proper version
                     Dim vers As String
50                   'vers = GetJDFVersion(xmldoc)
                     'If Not ((vers = JDFVersion10) Or (vers =
     JDFVersion11)) Then
```

```
'        MsgBox "The version of this JDF file (version:"
& vers & ") is not supported." & Chr(10) & "Use supplied templates."
& filepath, vbOKOnly + vbCritical
'                bLoadDefault = True
'          End If
          End If
      Else ' invalid document element
          MsgBox "Could not open JDF " & filepath, vbOKOnly +
vbCritical
          bLoadOK = False
      End If
    End If
    LoadJDF = bLoadOK
End Function


Private Sub Command1_Click()

End Sub


Rem Import JDF
' This function allows the user to select a JDF file (triggered by
clicking the ImportJDF button in the UI)
Private Sub ImportJDF_Click()
    Dim FileName As String

    ' InitXML
    Set xmldoc = Nothing
    Set xmldoc = CreateObject("Msxml.DomDocument")
    xmldoc.async = False

    Dim filepath As Variant
    Dim strFilter As String
    Dim lngFlags As Long

    ' Prepare Select File dialog box
    strFilter = ahtAddFilterItem(strFilter, "JDF Files (*.jdf)",
"*.jdf")
    strFilter = ahtAddFilterItem(strFilter, "All Files (*.*)",
"*.*")
    filepath = ahtCommonFileOpenSave(InitialDir:="", _
        Filter:=strFilter, FilterIndex:=1, flags:=lngFlags, _
        DialogTitle:="Open", hwnd:=Me.hwnd)

    ' Extract the data needed from the JDF file
    If (filepath <> vbNullString) Then
        FileName = filepath
        ' Load the file
        If LoadJDF(xmldoc, FileName) Then
          ' Extract the data and copy to locals
          Description.Text = GetJDFDescriptiveName(xmldoc)
          JobName.Text = GetJDFJobID(xmldoc)
          NrPages.Text = GetJDFIntentPages(xmldoc)
```

```
        JDFDimensions = GetJDFIntentDimensions(xmldoc)
        ' Convert to mms
        sPos = InStr(JDFDimensions, " ")
        Wstr = Left(JDFDimensions, sPos - 1)
        Hstr = Mid(JDFDimensions, sPos + 1)
        W = Round((CDbl(Wstr) / 72) * 25.4)
        PageWidth.Text = W
        H = Round((CDbl(Hstr) / 72) * 25.4)
        PageHeight.Text = H
    End If

End If
End Sub
```

Those skilled in the art will appreciate that numerous modifications and variations may be made to the embodiments disclosed above without departing from the scope of the present invention.

List of reference signs

10 : planning application

12 : product definition

5   14 : script

15 : template generator

17 : template

20 : layout application

22 : artwork

10

■